

- Secrets
- ABAP
- Apex
- C
- C++
- CloudFormation
- COBOL
- COBOL
- C#
- CSS
- Flex
- Go
- HTML
- Java
- JavaScript
- Kotlin
- Kubernetes
- Objective C
- PHP
- PL/I
- PL/SQL
- Python
- RPG
- Ruby
- Scala
- Swift
- Terraform
- Text
- TypeScript
- T-SQL
- VB.NET
- VB6
- XML



Flex static code analysis

Unique rules to find Bugs, Security Hotspots, and Code Smells in your FLEX code

All rules 76

Vulnerability 5

Bug 9

Security Hotspot 1

Code Smell 61

Tags

Search by name...

Security.allowDomain(...) should only be used in a tightly focused manner

Vulnerability

The flash.system.Security.exactSettings property should never be set to false

Vulnerability

Dynamic classes should not be used

Code Smell

"LocalConnection" should be configured to narrowly specify the domains with which local connections to other Flex application are allowed

Vulnerability

"default" clauses should be first or last

Code Smell

Event types should be defined in metadata tags

Code Smell

Event names should not be hardcoded in event listeners

Code Smell

The special "star" type should not be used

Code Smell

Variables of the "Object" type should not be used

Code Smell

Methods should not be empty

Code Smell

Constant names should comply with a naming convention

Code Smell

All branches in a conditional structure should not have exactly the same implementation

Bug

Switch cases should end with an unconditional "break" statement

Analyze your code

Code Smell Blocker cwe suspicious

When the execution is not explicitly terminated at the end of a switch case, it continues to execute the statements of the following case. While this is sometimes intentional, it often is a mistake which leads to unexpected behavior.

Noncompliant Code Example

```
switch (myVariable) {
    case 1:
        foo();
        break;
    case 2: // Both 'doSomething()' and 'doSomethingElse()'
        doSomething();
    default:
        doSomethingElse();
        break;
}
```

Compliant Solution

```
switch (myVariable) {
    case 1:
        foo();
        break;
    case 2:
        doSomething();
        break;
    default:
        doSomethingElse();
        break;
}
```










Exceptions

This rule is relaxed in the following cases:

```
switch (myVariable) {
    case 0: // Empty case used to specify the same behavior for multiple values
    case 1:
        doSomething();
        break;
    case 2: // Use of return statement
        return;
    case 3: // Use of throw statement
        throw new IllegalStateException();
    case 4: // Use of continue statement
        continue;
    default: // For the last case, use of break statement is not required
        doSomethingElse();
}
```

See

- MITRE, CWE-484 - Omitted Break Statement in Switch
- CERT, MSC17-C. - Finish every set of statements associated with a case label

<div>Classes that extend "Event" should override "Event.clone()"</div> <div> Bug</div>
<div>Constructors should not dispatch events</div> <div> Bug</div>
<div>"ManagedEvents" tags should have companion "Event" tags</div> <div> Bug</div>
<div>Objects should not be instantiated inside a loop</div> <div> Code Smell</div>
<div>Two branches in a conditional structure should not have exactly the same implementation</div> <div> Code Smell</div>
<div>Constructor bodies should be as lightweight as possible</div> <div> Code Smell</div>
<div>Only "while", "do" and "for" statements should be labelled</div> <div> Code Smell</div>
<div>Statements, operators and keywords specific to ActionScript 2 should not be used</div> <div> Code Smell</div>
<div>"for" loop stop conditions should be invariant</div> <div> Code Smell</div>
<div>Unused function parameters should be removed</div>

- with a break statement
- [CERT, MSC52-J](#) - Finish every set of statements associated with a case label with a break statement

Available In:

